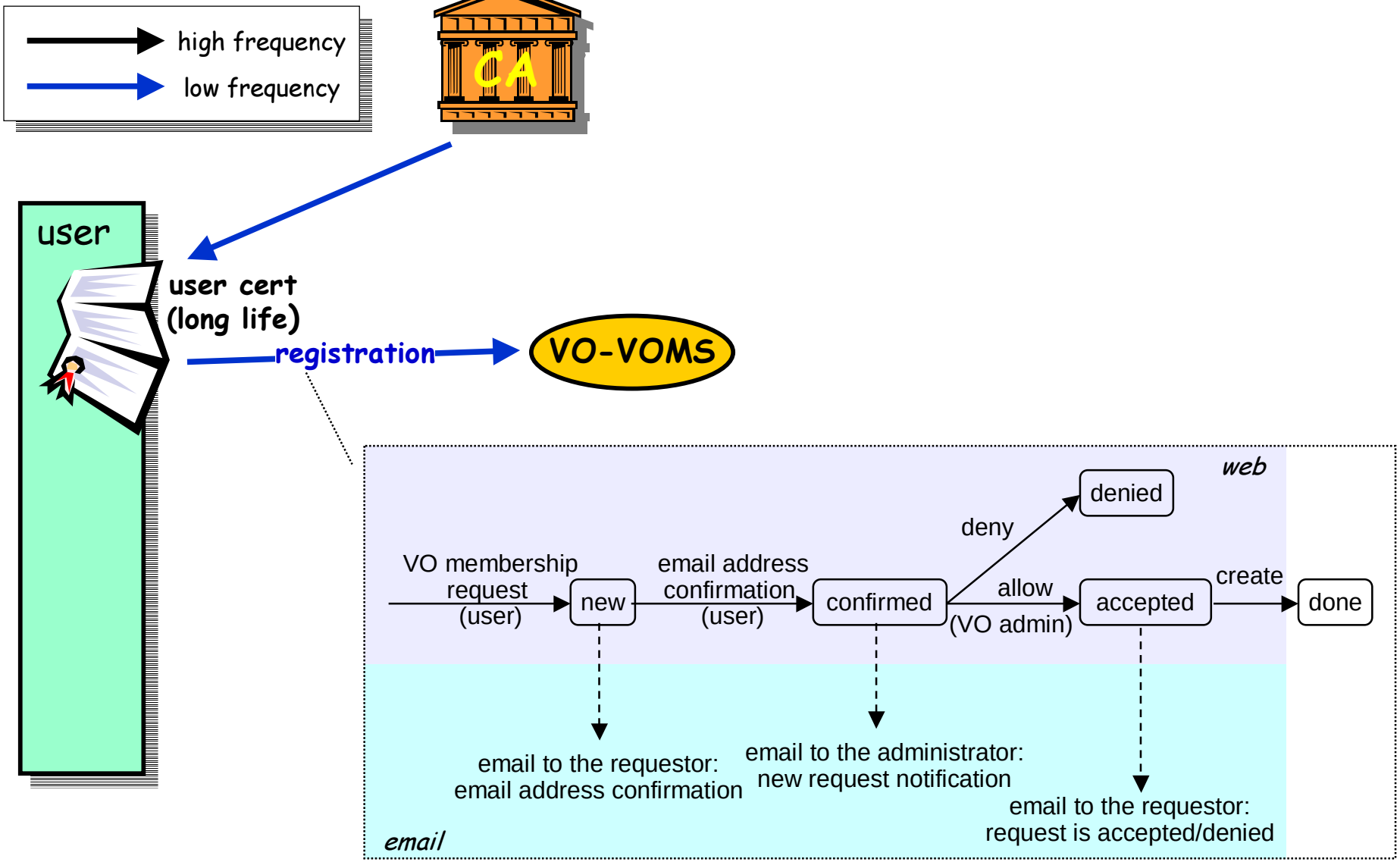
Update on EDG Security (VOMS)

European DataGrid Project

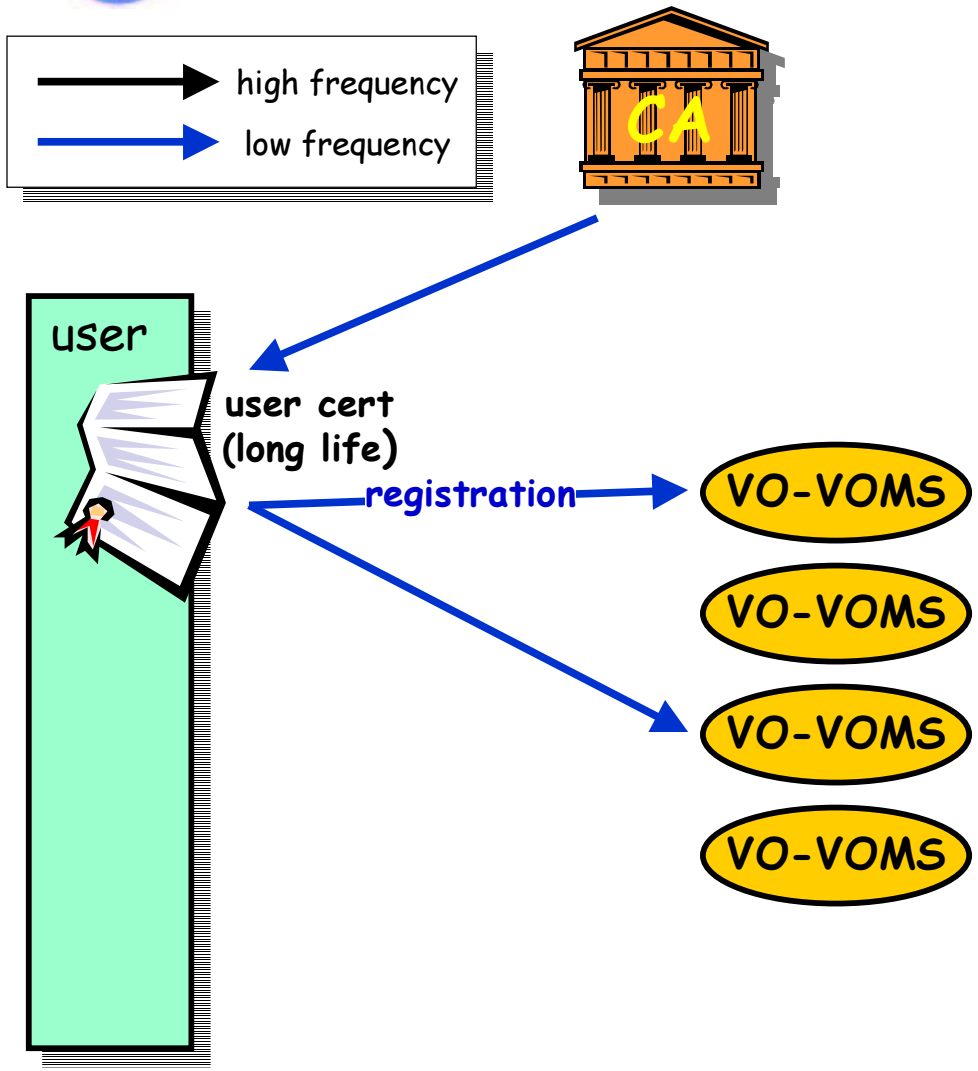
Security Coordination Group

http://cern.ch/hep-project-grid-scg

Registration



Multi-VO registration



VO administration operations

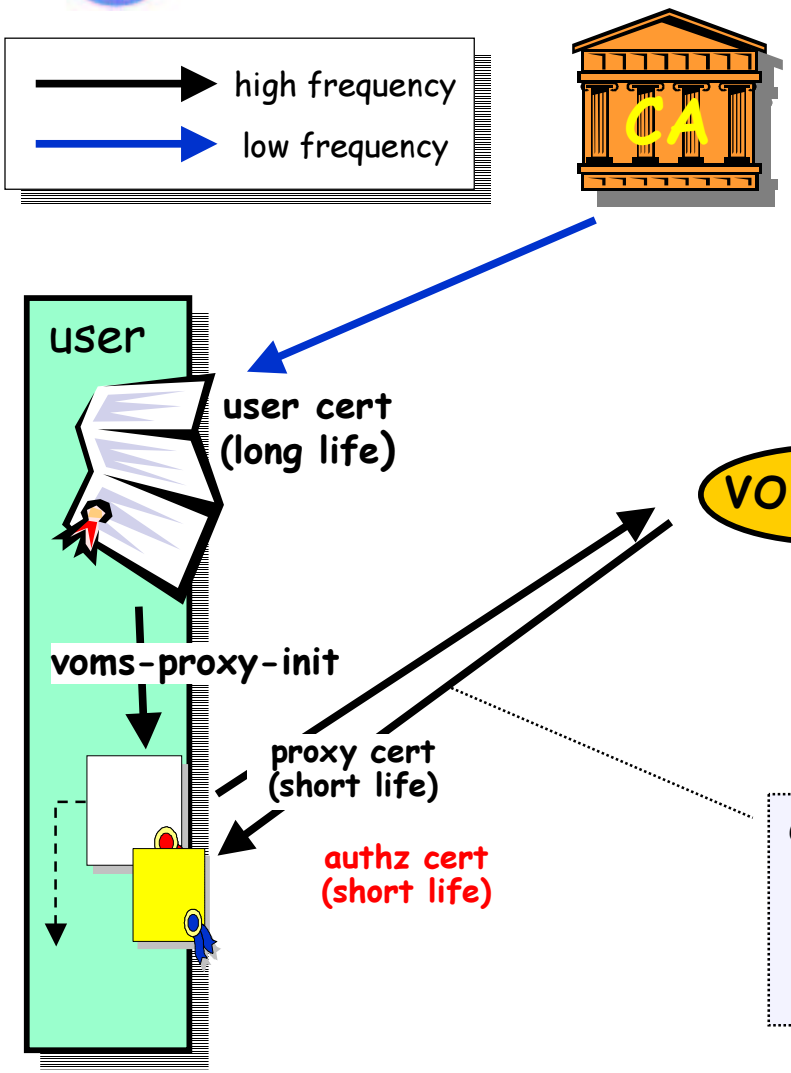
- create/delete (sub)group/role/capability
- add/remove member of g/r/c
- get/set ACLs for these operations

VO registration tasks

user requested administrative operation; e.g.:

user registration = add member

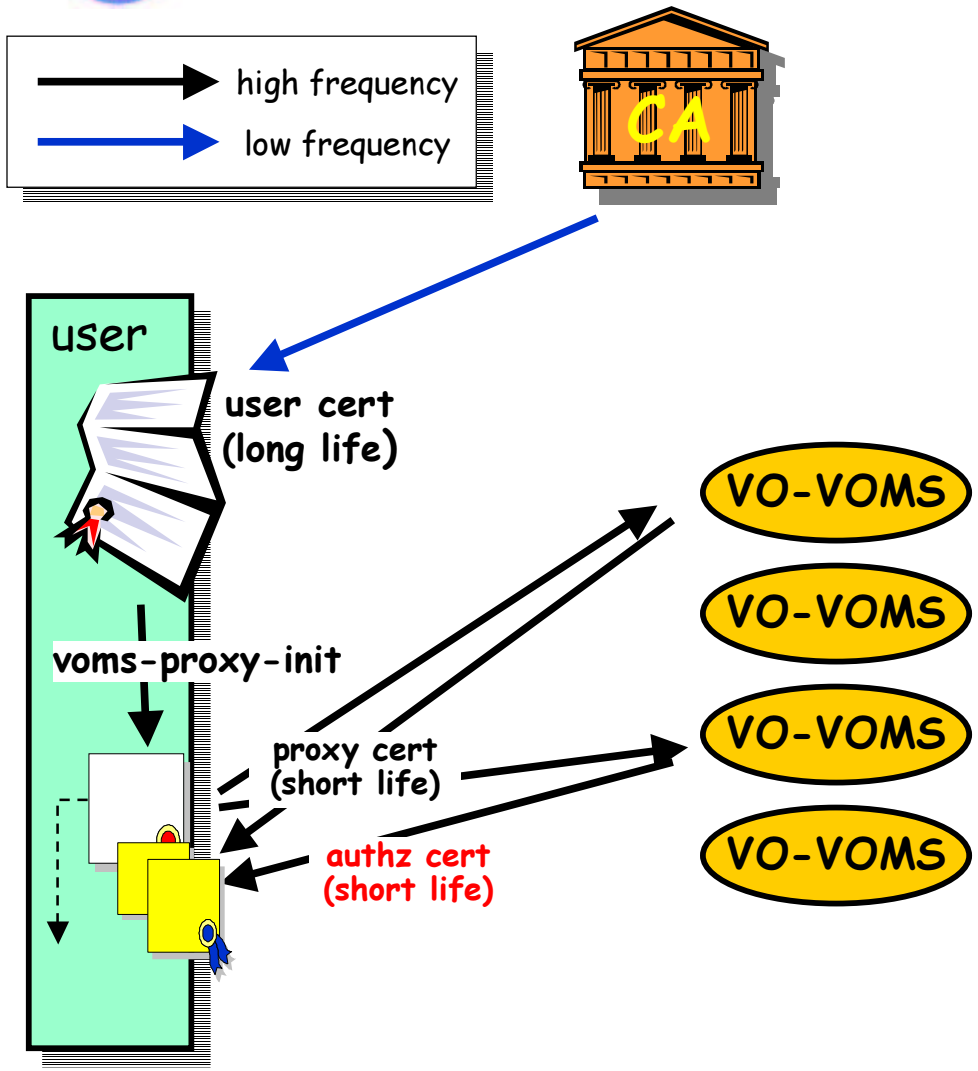
"Login"



edg-voms-proxy-init -voms iteam

- /tmp/x509_up<UID> (normal proxy location)
- backward compatible proxy format

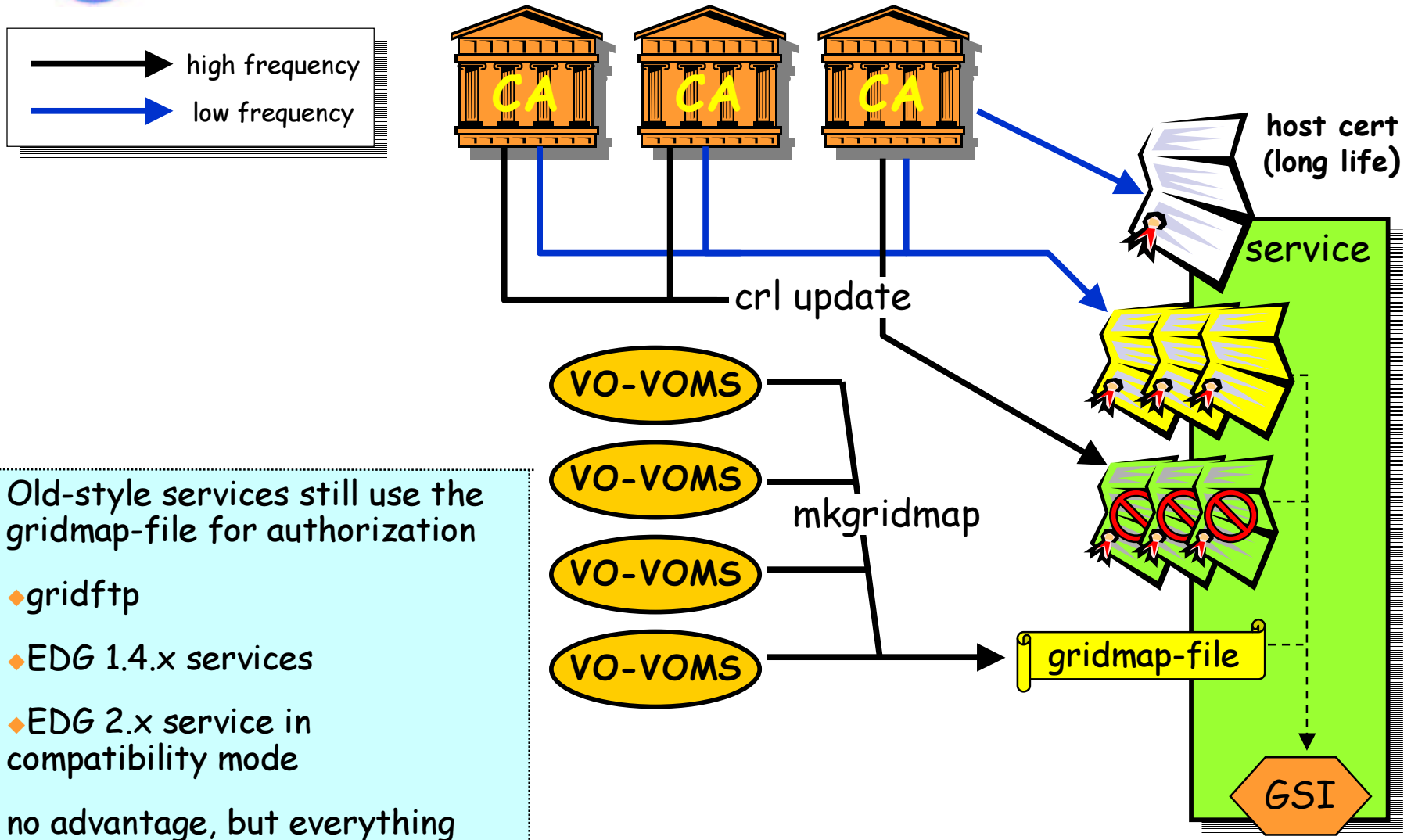
Multi-VO "Login"



voms-proxy-init -voms iteam -voms wp6

- single proxy certificate is generated
- each VO provides a separate VOMS credential
  first one is the default VO
- each VOMS credential contains multiple group/role entries
  first one is the default group

Old-style Service

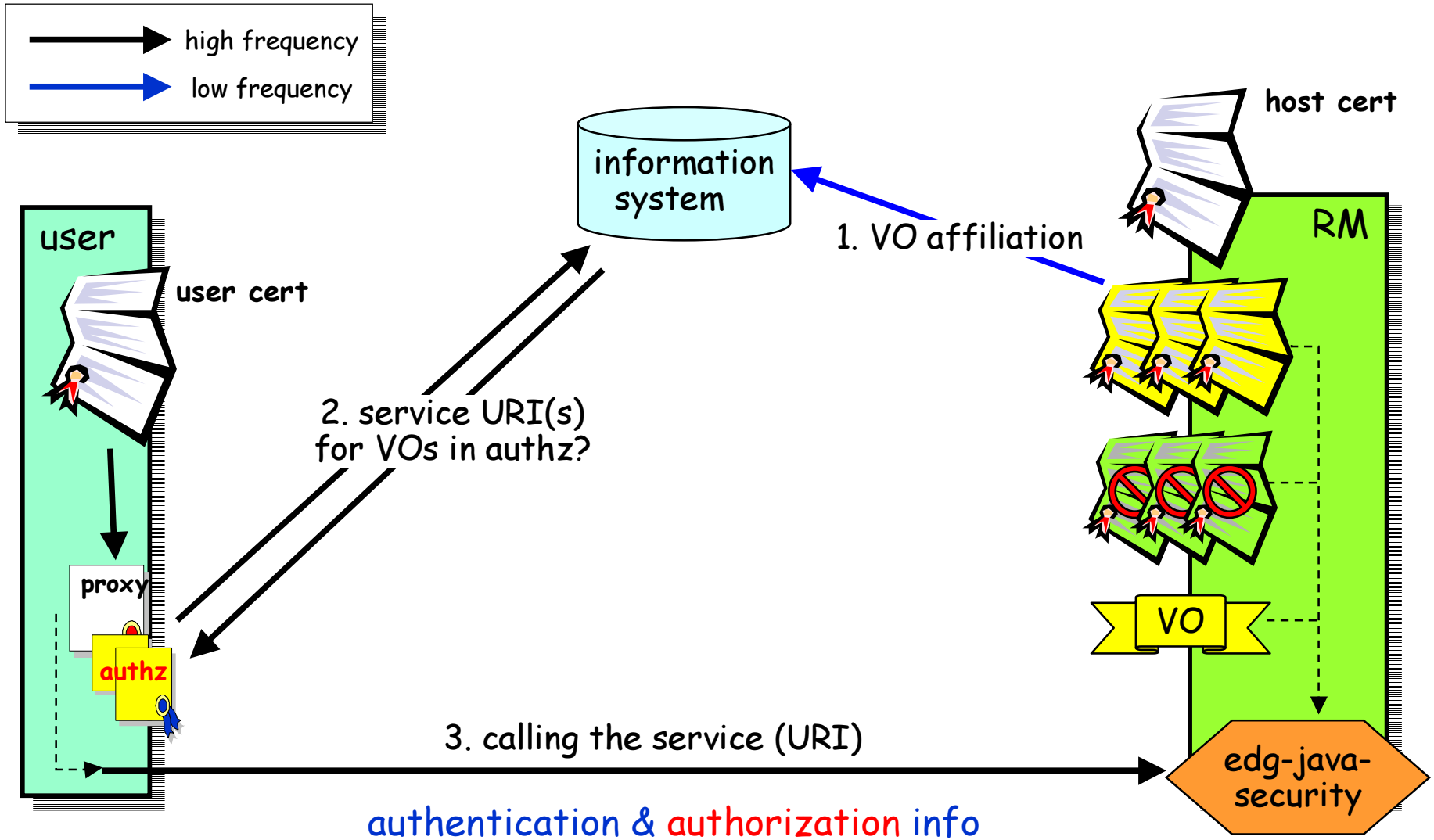


Old-style services still use the gridmap-file for authorization

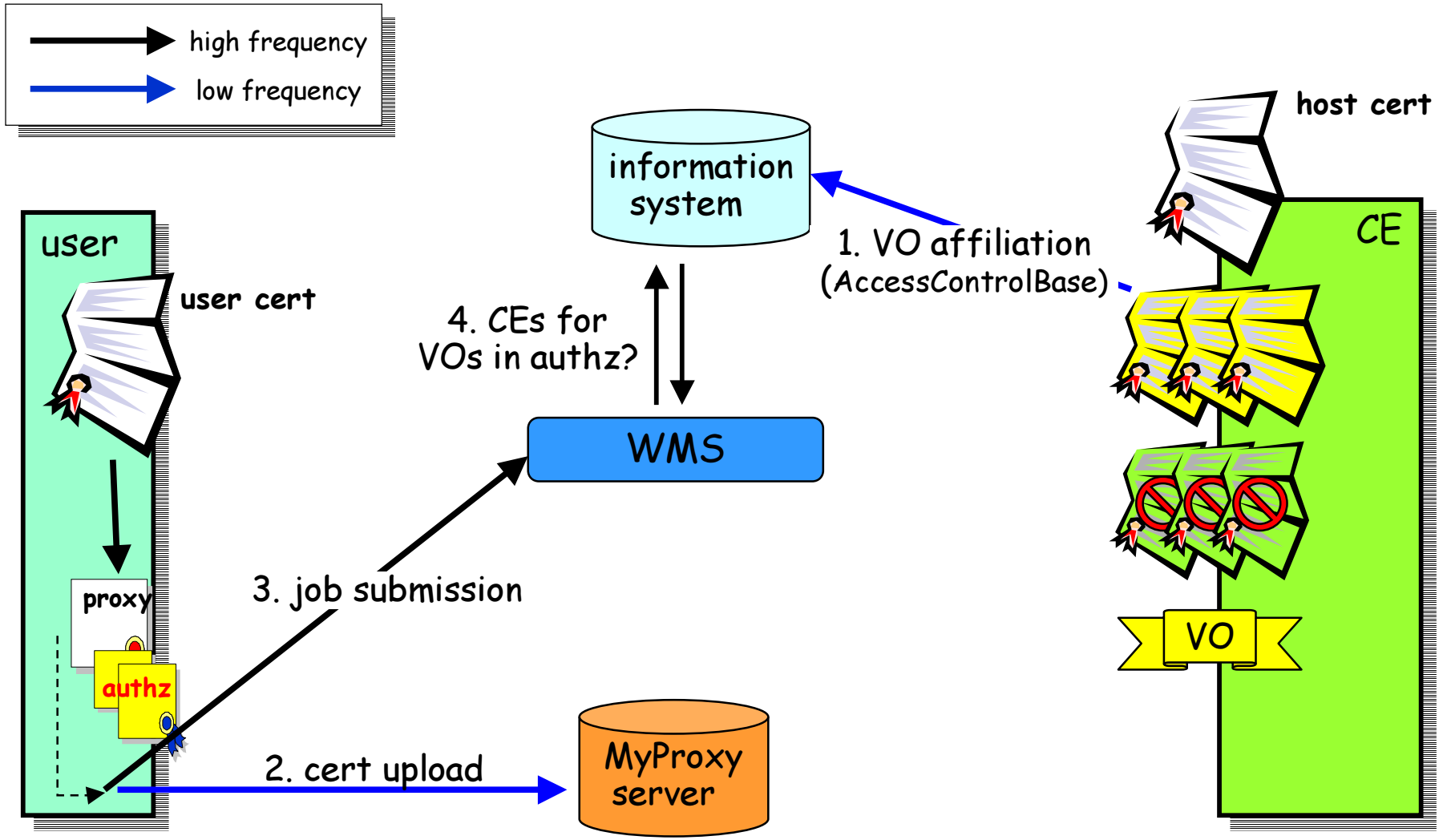
- gridftp
- EDG 1.4.x services
- EDG 2.x service in compatibility mode

no advantage, but everything works as before...

Replica Management



Job Submission

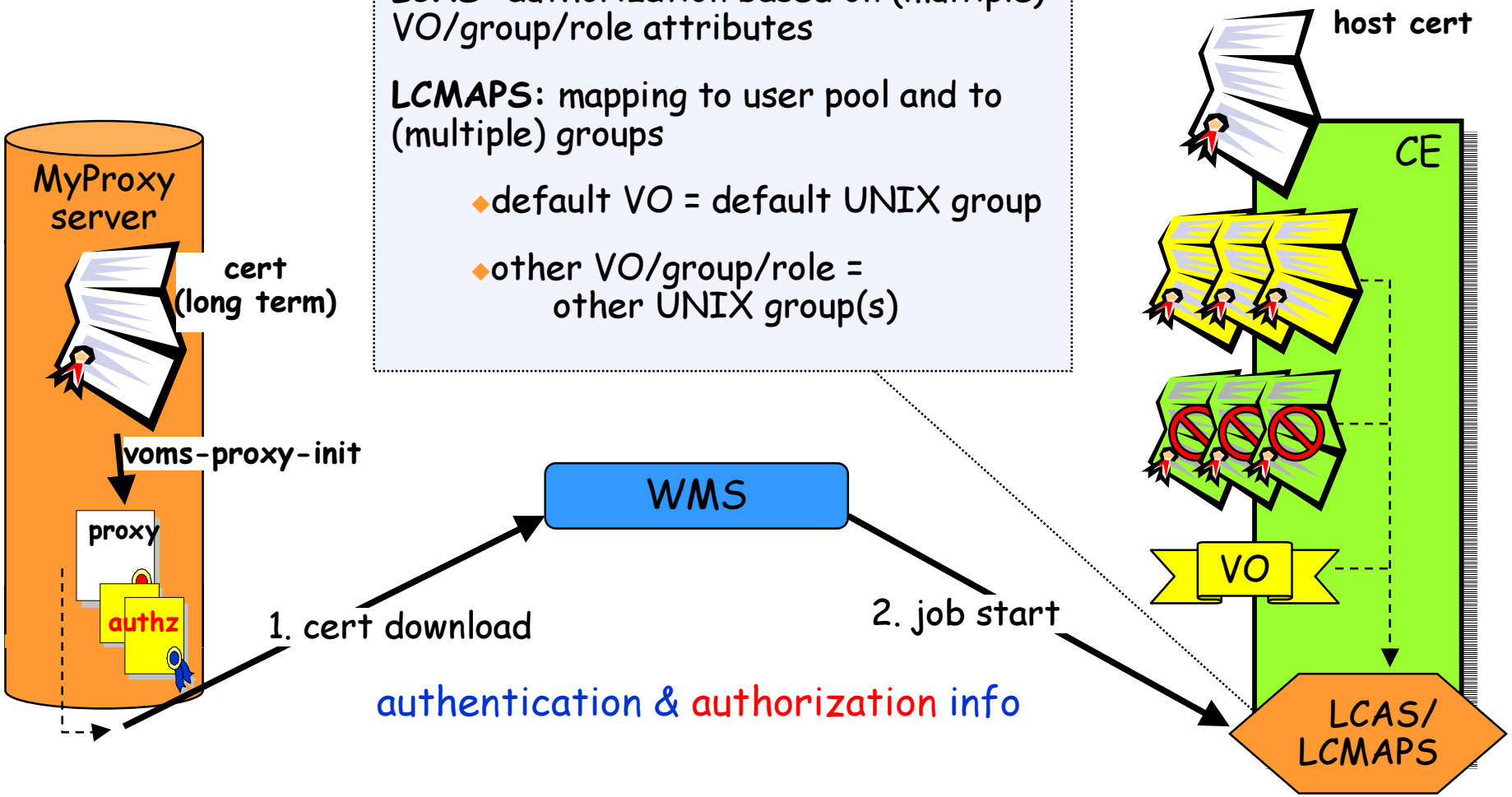


Running a Job

LCAS: authorization based on (multiple) VO/group/role attributes

LCMAPS: mapping to user pool and to (multiple) groups

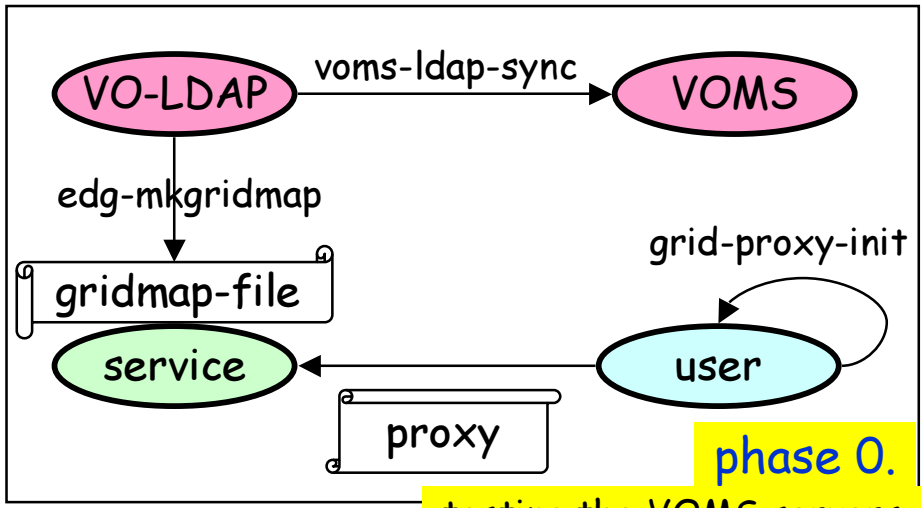
- default VO = default UNIX group
- other VO/group/role = other UNIX group(s)

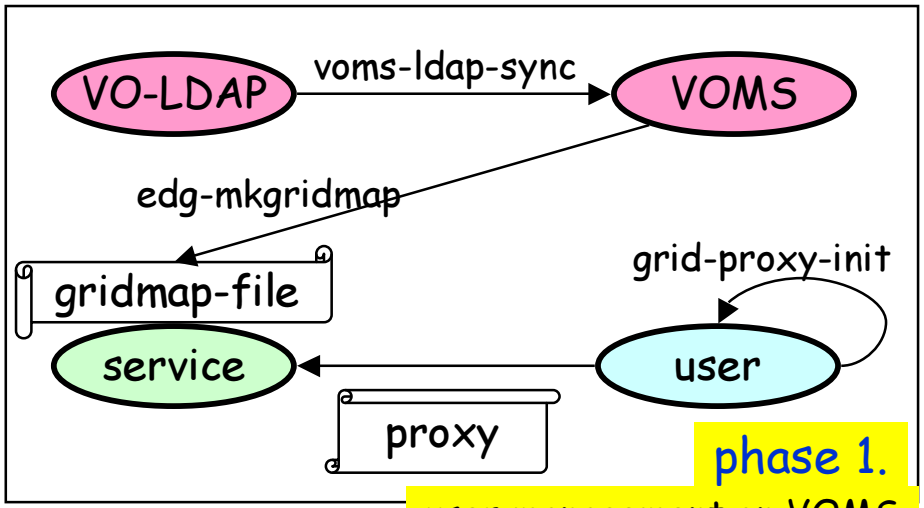


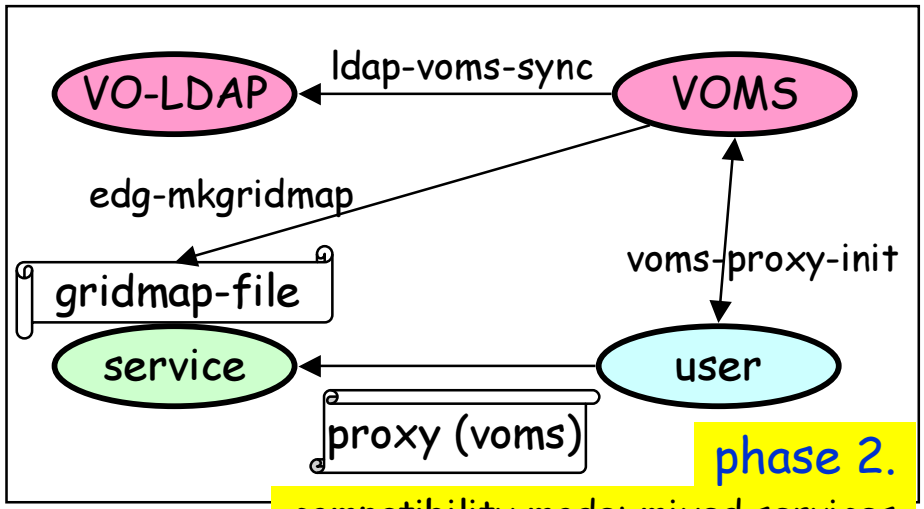
VOMS FAQ

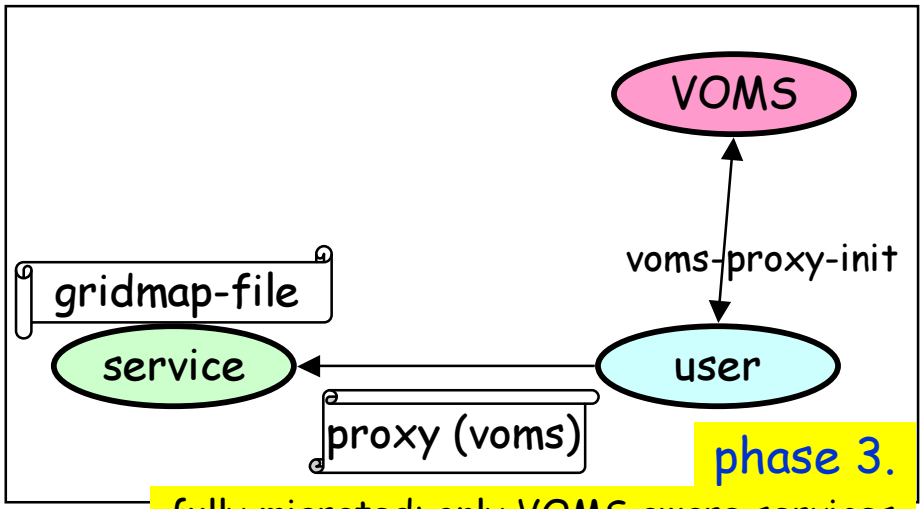
- No instant effect: the user has to "log-in", using voms-proxy-init, to be notified of any VO change
- Delegation: a user cannot delegate her/his groups to someone else (unless s/he is a group-admin); no user groups
- Indirect effect on the policy: VOMS may name groups/roles in order to implement a policy, but it is up to the services to enforce it and up to the resource owner no to override it
- VOMS is not used to implement fine grained ACLs: it does not store file names or job ids (although it has its *own* ACLs for group/role administration)

VOMS migration









Auth/Authz in Services

- GSI based or compatible authentication
- grid-mapfile or VOMS based authorization (can be both)
- policy or ACL based access control
  - coarse and fine grained solutions
  - access control description's syntax is not standard
- implemented alternatives:
  - edg-java-security for Java web services
  - GSI/LCAS/LCMAPS for native C/C++ services
  - mod_ssl/GACL for Apache based web services
  - (Slahgrid for transparent filesystem ACLs)

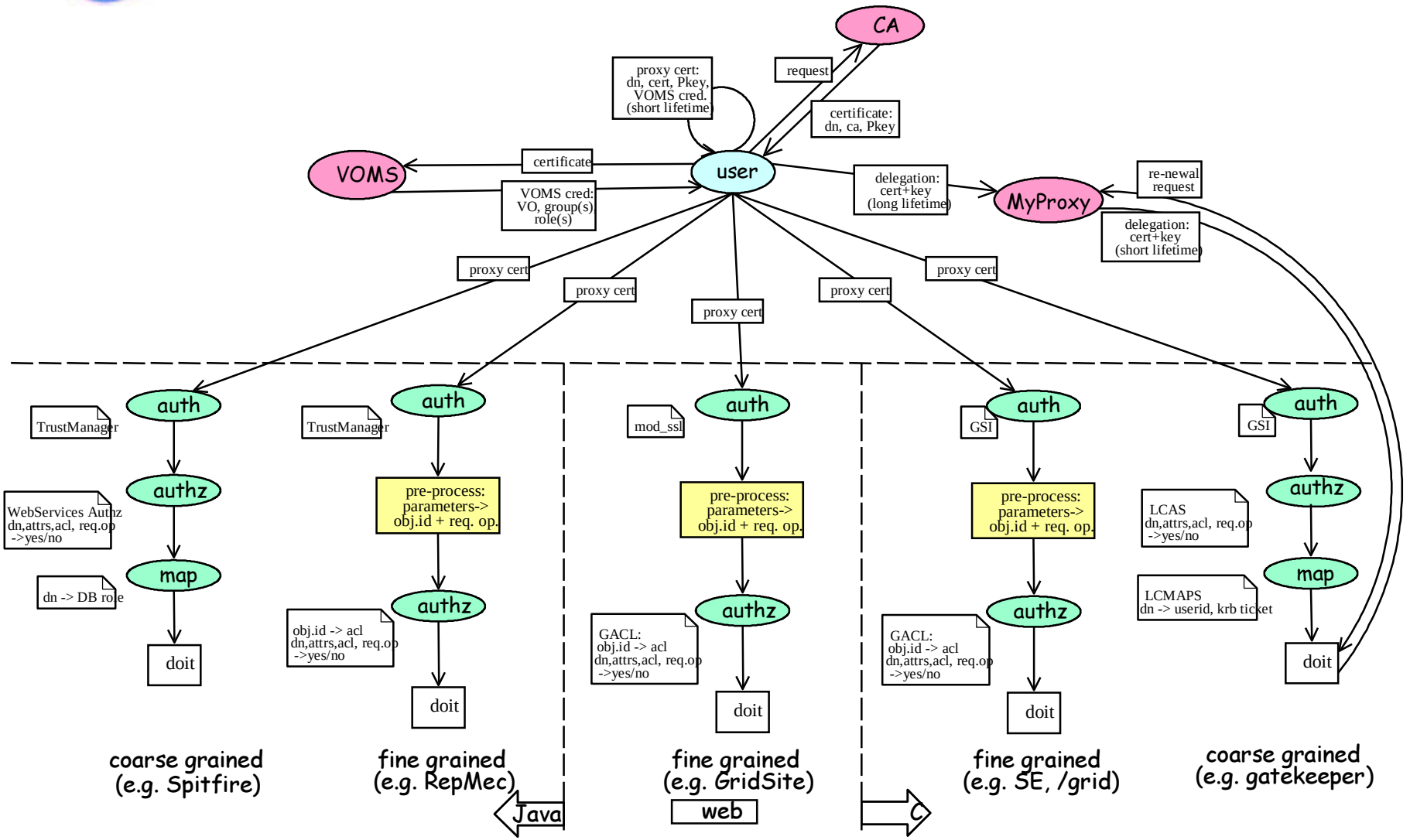
Local Site Authorization

- Local Centre Authorization Service (**LCAS**)
  - Handles authorization requests to local fabric
    - authorization decisions based on proxy user certificate and job specification;
    - supports *grid-mapfile* mechanism.
  - Plug-in framework (hooks for external authorization plugins)
    - allowed users (*grid-mapfile* or *allowed_users.db*), banned users (*ban_users.db*), available timeslots (*timeslots.db*)
    - plugin for VOMS (to process authorization data)
- Local Credential Mapping Service (**LCMAPS**)
  - provides local credentials needed for jobs in fabric
  - mapping based on user identity, VO affiliation, local site policy

edg-java-security

- Trust manager
  - GSI compatible authentication
  - Adapters to HTTP and SOAP
  - Currently deployed for Tomcat4
- Authorization Manager
  - Authorization and mapping for Java services
  - Plug-in framework for maps: database, XML file and for backward compatibility: gridmap-file
  - Handles VOMS attributes

Overview


CA
request
proxy cert:
dn, cert, Pkey,
VOMS cred.
(short lifetime)
certificate:
dn, ca, Pkey
VOMS
certificate
user
VOMS cred:
VO, group(s),
role(s)
delegation:
cert+key
(long lifetime)
MyProxy
re-newal
request
delegation:
cert+key
(short lifetime)
proxy cert
proxy cert
proxy cert
proxy cert
proxy cert
auth
TrustManager
authz
WebServices Authz
dn,attrs,acl, req.op
->yes/no
map
dn -> DB role
doit
auth
TrustManager
pre-process:
parameters->
obj.id + req. op.
authz
obj.id -> acl
dn,attrs,acl, req.op
->yes/no
doit
auth
mod_ssl
pre-process:
parameters->
obj.id + req. op.
authz
GACL:
obj.id -> acl
dn,attrs,acl, req.op
->yes/no
doit
auth
GSI
pre-process:
parameters->
obj.id + req. op.
authz
GACL:
obj.id -> acl
dn,attrs,acl, req.op
->yes/no
doit
auth
GSI
authz
LCAS
dn,attrs,acl, req.op
->yes/no
map
LCMAPS
dn -> userid, krb ticket
doit
coarse grained
(e.g. Spitfire)
fine grained
(e.g. RepMec)
Java
fine grained
(e.g. GridSite)
web
C
fine grained
(e.g. SE, /grid)
coarse grained
(e.g. gatekeeper)